5. Contents of the tape: The submitted tape contains four files optionally in single or double precision FORTRAN.

NLP: This file contains the nonlinear programming algorithm in form of a main program, where the data are read in, or in form of two subroutines called NLPQL and NLPQL1, respectively. Their usage is completely explained by the initial comments. Note that the code does not contain a subroutine to solve the quadratic programming or, optionally, the linear least squares program. A frame called QL is included in NLP to adapt the quadratic programming subroutine QPSOL of Gill, Murray, Saunders, and Wright [1], and has to be modified, if any other algorithm is to be implemented.

TEST: The file consists of a main program to execute NLPQL on the test problems published by Hock and Schittkowski [2], and of some auxiliary routines.

LS: The file contains some subroutines to adapt the linear least squares program NNLS published in Lawson and Hanson [5], which could be implemented alternatively to solve the subproblem. In this case, the user has to implement NNLS and its auxiliary subroutines H12, G1, and G2 in addition, and he should scratch subroutine QL from the file NLP.

PROB: The previously mentioned 115 test examples of Hock and Schittkowski have been coded by the authors together with their derivatives. If the test frame contained in the file TEST is to be executed, then combine certain groups of test examples in subfiles as indicated by some linking subroutines called CONV.

6. Literature: The mathematical algorithm, its convergence properties, the usage of the programs, extensive test results, and more implementation details are published in [10]. The mathematical algorithm is described in [9] and some numerical results of a preliminary version based on least squares subproblems are found in [8].

7. Numerical tests: The program NLPQL has been tested extensively on approximately 500 test problems. Different versions of the code obtained by exchanging program modules, have been executed on the hand-selected and real life test examples of Hock and Schittkowski [3]. Furthermore the numerical performance of NLPQL has been evaluated on randomly generated optimization problems as described in [7] and can be compared with the performance of 26 other available nonlinear programming codes. The results are published in [10]. There have been some practical applications of preliminary versions of NLPQL. Among those are a taxation problem at the University of Rotterdam, cf. Louter and Stern [6], and the optimal velocity control of a cryogenic cooled wind canal at the DFVLR in Oberpfaffenhofen, cf. Kraft [4]. Further copies of NLPQL have been purchased by several mechanical engineering departments to solve mechanical structural design problems.

8. Hardware facilities: The numerical tests cited in section 7 have been performed on an IBM 370/168 at the Stanford University in double precision and on a Telefunken TR440 at the University of Würzburg in single precision arithmetic.

9. Requests of the author: Although the author feels that the programs have been tested as thoroughly as possible, no guarantee can be given that the codes are completely free of errors. The user is kindly requested to inform him about programming errors, difficulties in implementing and compiling the FORTRAN source, possible misinterpretations of the documentation, and any features that could improve the performance and usage of the codes. In addition the author would be very grateful to learn about practical applications of nonlinear programming algorithms and asks the user to send him any available information material (preprints, computer outputs, etc.).

10. Availability: For more information about the mathematical algorithm or the conditions for obtaining the program package on a magnetic tape, write to

Klaus Schittkowski
Institut für Angewandte Mathematik und Statistik
Universität Würzburg
D-8700 Würzburg, Germany, F.R.

An order form is available to specify the desired magnetic tape format. The Institut für Angewandte Mathematik und Statistik requires a charge of DM 250.- (approximately $100.-) for the handling costs.

REFERENCES

[1] P.E. Gill, W. Murray, M.A. Saunders, M.H. Wright, User's guide for SOL/QPSOL: A FORTRAN package for quadratic programming, Technical Report SOL 82-7, Dept of Operations Research, Stanford University, Stanford, USA, 1982.

[2] W. Hock, K. Schittkowski, Test examples for nonlinear programming codes, Lecture Notes in Economics and Mathematical Systems, Vol.187, Springer, Berlin, Heidelberg, New York, 1981.

[3] W. Hock, K. Schittkowski, Comparative performance evaluation of 27 nonlinear programming codes, to appear: Computing.

[4] D. Kraft, Optimal velocity control of a cryogenic cooled wind canal, to appear: DFVLR-Report, DFVLR Oberpfaffenhofen, Germany, F.R.

[5] C.L. Lawson, R.J. Hanson, Solving least squares problems, Prentice Hall, Englewood Cliffs, New Jersey, 1974.

[6] A.S. Louter, V. Stern, Household behaviour under taxation, Report 8132/E, Econometric Institute, Erasmus University, Rotterdam, Netherlands, 1981.

[7] K. Schittkowski, Nonlinear programming codes, Lecture Notes in Economics and Mathematical Systems, Vol.183, Springer, Berlin, Heidelberg, New York, 1980.

[8] K. Schittkowski, The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function. Part 2: An efficient implementation with linear least squares subproblems, Numerische Mathematik, Vol.38 (1981), 115-127.

h) Restart in error cases: **If requested by a user, NLPQL will** perform automatic restarts in error situations, cf. k).

i) Modular structure: NLPQL possesses a modular structure. Besides the solution method for the subproblem, a user could replace any of the default subalgorithms, i.e. he could replace the merit function, the steplength algorithm, and the numerical differentiation routine. The usage of these modules is described in the documentation, cf.[8]. In particular he could use NLPQL as a frame to test one of the mentioned modules. This is facilitated by the inclusion of a complete test program and a set of 115 nonlinear programming test examples,cf.l).

j) Output facilities: A user is allowed to suppress all output. Alternatively a final convergence analysis or additional output for each iteration step can be required. Some further output from the program modules mentioned above can be printed on request.

k) Error analysis: A detailed error analysis shows the user the termination reason of the algorithm if the problem could not be solved successfully. Besides failures which could occur in a user provided module, the algorithm will stop in one of the following error cases: Upper bound of iteration number reached, uphill search direction, underflow in BFGS-update, upper bound for line search iteration number reached, length of a working array too short, false dimensions, etc.

l) Test program: The program package distributed by the author, contains a complete test program and a set of 115 test problems, cf.[7]. A user is able then to test the correct implementation of NLPQL and to compare his results with those obtained by the author.

numerical performance of NLPQL. In this case, the positive definite matrix describing the quadratic approximation of the Lagrangian function, will be replaced by its triangular factor. A user could exploit this feature for solving unconstrained or simply constrained optimization problems.

c) Feasible starting point for the subproblem: In certain error situations, the subproblem will be expanded by an additional variable to avoid inconsistent linearized constraints. Alternatively the expanded subproblem could be formulated in each iteration, so that the quadratic programming or linear least squares algorithm is always provided with a feasible starting point.

d) Scaling: An automatic scaling procedure is included in NLPQL based on a default value. This specific scaling factor can be omitted or altered. It is furthermore possible to provide NLPQL with individual scaling values for objective and constraint functions when executing the code.

e) Reverse communication: A user is allowed to perform reverse communication, the most flexible way to solve an optimization problem. In this case, only one iteration step will be performed by NLPQL. Then the subroutine returns to the main program where new function and gradient values have to be calculated. A subsequent call of NLPQL continues the iteration.

f) Numerical differentiation: If a user wants to calculate the gradients analytically, he has to provide a subroutine called GRAD. Otherwise a subroutine included in the program package will be executed automatically, utilizing a simple forward difference formula.

g) Additional problem information: Initially the approximation matrix for the Hessian is set to the identity matrix and the initial estimates for the multipliers are set to zero. Alternatively a user could provide NLPQL with his own guesses.

[9] K. Schittkowski, On the convergence of a sequential quadratic programming method with an augmented Lagrangian line search function, Technical Report SOL 82-4, Dept. of Operations Research, Stanford University, Stanford, USA, 1982.

[10] K. Schittkowski, Design, implementation, and test of a nonlinear programming algorithm, to appear.

An Expected Utility Approach for Ranking Heuristics and Other Alternatives

by

B. Golden, A. Assad, E. Wasil
University of Maryland at College Park

Consider a decision-maker who is in possession of several different methods or techniques for attacking a given class of decision problems. The alternative methods share a common objective but, being different, produce different results upon application to a particular problem. Given an arsenal of alternative methods a natural question is: Which one should be chosen to achieve the "best" results overall?

In the two Working Papers cited below, we provide a methodological procedure for choosing the best technique from a number of alternatives based on their performance over a battery of typical problems. The proposed expected utility approach was originally developed in connection with the comparison of heuristic procedures for combinatorial optimization problems.

Suppose that a decision-maker has available K alternative methods A1, A2,..., AK and that a set of n test problems P_1, $P_2,\dots, P_n$ are chosen on which to apply the K methods. We let X_{ij} denote the results of applying alternative Aj to the i-th test problem. Since there are serious drawbacks to analyzing the X_{ij} values directly, the results are modified to report the relative accuracy of each result in terms of a percentage deviation from a reference point known as the target value t_i for the i-th problem. For example, if the decision-maker is studying the performance of K heuristic procedures with respect to a set of n instances of a specific combinatorial minimization problem, then X_{ij} would represent a heuristic solution and the target value might be a known lower bound on the optimal solution to the problem. Thus, the percentage deviation P_{ij} (of the result obtained by method j on the i-th problem) from the target value t_i would be defined as

$$P_{ij} = 100\left(\frac{X_{ij}}{t_i} - 1\right) \qquad 1 \le i \le n;\ 1 \le j \le K.$$

After the decision-maker has replaced the X_{ij} values with the corresponding set of P_{ij}'s, the expected utility approach would be used to provide a ranking of the alternative methods. The approach models the performance of each method probabilistically in terms of a gamma distribution and chooses a risk-averse

a user adapts an available library routine or tries to implement one of the proposed algorithms, cf.[7]. The problem size is therefore limited by the core size and the capability of the quadratic programming subroutine to solve large problems. Note also that the accuracy and reliability of the quadratic programming algorithm could influence the overall numerical performance of NLPQL significantly. The nonlinear programming code has been tested successfully on small size, smooth, and deterministic problems, i.e. on continuously differentiable problems with $m,n \leqslant 50$, where all functions could be evaluated within the machine precision, cf.[9]. For this type of optimization problems, the algorithm seems to be quite robust with respect to numerical differentiation.

4. Capabilities of NLPQL: NLPQL possesses the following capabilities for facilitating the solution of a problem or to adapt the algorithm to a specific situation.

a) Different execution levels: The numerical solution of a nonlinear programming problem can be performed in three different ways. First it is possible to use a main program where only the most important data are read in, i.e. n, m_e, m, IPRINT, x_l, x_o, x_u . IPRINT controls the desired output and the array x_o has to contain a starting point. Alternatively, an easy-to-use subroutine NLPQL may be executed, where the same data and some working arrays have to be declared in form of subroutine arguments. To alter default values or to solve the problem in one of the ways outlined subsequently, the flexible and problem adaptable subroutine NLPQL1 is offered.

b) Alternate subproblems: Instead of the quadratic programming subproblem, it is possible to formulate a linear least squares problem in each iteration. This allows the use of any available least squares program instead of a quadratic programming routine. In principal, any carefully implemented linear least squares algorithm will lead to the same

1. Mathematical model: The program NLPQL has been designed to solve the general nonlinear programming problem

$$
\begin{array}{ll}
 & \min \; f(x) \\
 & g_j(x) = 0 \quad , \; j=1,\ldots,m_e \quad , \\
x \in R^n : & g_j(x) \geqslant 0 \quad , \; j=m_e+1,\ldots,m \quad , \\
 & x_l \leqslant x \leqslant x_u \quad .
\end{array}
$$

It is assumed that all problem functions are continuously differentiable. Linear constraints cannot be provided separately, but the algorithm will take advantage of their special simple structure implicitly, cf. 2 .

2. The algorithm: NLPQL utilizes a sequential quadratic programming method, also called recursive quadratic programming, variable metric, or Wilson, Han, Powell algorithm. In each iteration, a quadratic programming subproblem is formulated by linearizing the constraints and approximating the corresponding Lagrangian function quadratically. A new iterate is calculated then by minimizing a so-called merit or penalty function along the search direction obtained from the subproblem. The Hessian matrix of the Lagrange function is approximated by the BFGS-method, i.e. a special quasi-Newton or variable metric update formula. If some or all constraints are linear, then the algorithm provides the subproblem with the same linear constraints. In particular the solution method for the subproblem which is to be implemented by the user, can take advantage of special structures.

3. Domain of application: The program requires core storage for approximately $n^2 + mn + 9m + 28n$ real variables plus the additional storage which might be required to solve the quadratic programming subproblem. Note that a subroutine for solving that subproblem, is not included in the program package distributed by the author. The intention is that

decreasing utility function to reflect the decision-maker's attitude toward performance results. The methods are then ranked according to their expected utility values.

Briefly, the expected utility approach involves the following steps:

Step 1 Choose a target value t_i for each problem and compute the (absolute) percentage deviation of the results of applying alternative methods from the target value.

Step 2 Fit a probability distribution to the observed percentage deviations of a single method across the various test problems.

Step 3 Select a fixed utility function and calculate the expected utility for each alternative using the distribution obtained in Step 2.

Step 4 Rank order the alternatives according to their expected utility values and select the alternative with the largest value as the preferred method.

In Step 2, we have chosen a two-parameter gamma distribution of the form

$$h(x) = \left(\frac{x}{b}\right)^{c-1} e^{-x/b}/b\Gamma(c).$$

However, if there are some $p_{ij} = 0$, then a mixed gamma distribution of the form

$$f(x) = \rho\delta_0(x) + (1-\rho)\,h(x) \qquad 0 \le \rho \le 1$$

where $\delta_0(x)$ is the delta function based at 0 and $h(x)$ is a two-parameter gamma distribution might be more appropriate.

In Step 3 a decreasing utility function which is either increasingly, constantly, or decreasingly risk averse may be chosen.

The expected utility approach outlined above has been applied to studies in such diverse fields as combinatorial optimization, economics, operations management, marketing, education, and forecasting. In each of these applications the ranking obtained via the expected utility approach generally confirmed the original researcher's findings and also provided additional information in the form of a complete ranking of all alternatives.

References

1. B. Golden and A. Assad, "An Analytic Framework for Comparing Heuristics," Management Science & Statistics Working Paper 82-002, University of Maryland (1982).

2. E. Wasil, B. Golden, and A. Assad, "Diverse Applications of an Expected Utility Comparison Approach," Management Science & Statistics Working Paper 82-031, University of Maryland (1982).

Some implementation details for a nonlinear programming algorithm

- Klaus Schittkowski -
Institut für Angewandte Mathematik und Statistik
Universität Würzburg
8700 Würzburg
Germany, F.R.

The idea to implement the nonlinear programming algorithm considered here was conceived after finishing a comparative study of 26 optimization codes, cf. [3,7]. These FORTRAN programs were submitted in original form by their authors and have been tested extensively on about 500 randomly generated, hand selected, and real life test problems. The computational tests established the efficiency of the successive quadratic programming algorithms. To investigate the performance of these algorithms in more detail and to understand their outstanding behaviour, the author began to implement and test his own version, called NLPQL. This code was not designed to improve upon the efficiency of the best programs of the comparative study, e.g. Powell's program VFO2AD, since it uses the same mathematical strategy, but rather to implement an efficient, reliable algorithm in an user oriented, flexible, and modular way. This note describes the organization of the program, which is available now on request from the author.

AN MPS PRIZE FOR EXCELLENCE IN COMPUTATIONAL MATHEMATICAL PROGRAMMING?

J.K. Lenstra
Mathematisch Centrum, Amsterdam

A.H.G. Rinnooy Kan
Erasmus University, Rotterdam

Should the COAL proposal for the creation of an MPS Prize for Excellence in Computational Mathematical Programming be accepted? With due respect for the contributions made by COAL in the past, we have strong doubts on this issue. This is not because we would want to deny the importance of computational studies. On the contrary, the ultimate test for mathematical programming algorithms is how they behave in practice, and the proper design and analysis of experiments to predict this practical performance is of evident interest. However, so are many other aspects of our profession. Why should this particular activity then be singled out for a special prize?

It is not as though no other prizes are available to honor excellent computational work. For example, the chairman of the Dantzig Prize Committee has confirmed that this prize might well be handed out for research centering around computations. Recently Harlan Crowder, Martin Grötschel and Manfred Padberg received honorable mentions at the occasion of the 1980 ORSA Lanchester Prize for research on the traveling salesman problem that had a strong computational flavor. And at the TIMS XXVI International Meeting (June 1984, Copenhagen), a $2000 IBM prize will be awarded for the best paper showing the role of computers in operations research. There is nothing in the rules governing these and other prizes that disqualifies computational work. In fact, an unfortunate side effect of a separate prize would probably be to cause such a disqualification, simply because of its existence.

More generally, it does not seem to make sense to create a separate prize for a relatively small subarea emphasizing empirical tests and evaluations. This might even increase the isolation of the area and reinforce the lack of appreciation under which experimental research was carried out for so long, in the same way that specialized journals sometimes add to the splendid isolation in which certain fields of science choose to develop themselves. Computational studies form an integral part of mathematical programming, and its practitioners should be as much part of the mainstream of the Mathematical Programming Society as the other categories of researchers that compete for the existing prizes. If the COAL proposal would be accepted, what is there to stop other small groups from advancing similar proposals? Are we to have a GOAL prize for multicriteria optimization, a GHOST prize for fuzzy investigations and another COAL prize for energy studies? If anything, there are too many prizes already rather than too few.

The Society should certainly encourage good computational work. It should stimulate its publication in *Mathematical Programming* and in the *Mathematical Programming Studies*, it should provide facilities for the Committee on Algorithms, it should welcome special sessions at the Symposia and it should sponsor special meetings. Real recognition for excellence in computational mathematical programming will occur only when a prominent researcher from this area receives one of the available professional prizes in direct competition with excellent researchers from other areas. This has happened in the past, and there is no reason to believe that such an incentive will be less effective in the future.

REDUNDANCY IN MATHEMATICAL PROGRAMMING

Mark H. Karwan, Vahid Lotfi, Stanley Zionts
State University of New York at Buffalo

and

Jan Telgen
Rabo Bank
Zeist, The Netherlands

The authors of this paper conducted a study during 1980 and 1981, investigating the efficiency of size-reduction techniques in linear programming (LP) problems. The study, as well as new methods which were developed as a consequence, are published in [1]. In this note we present a general description of the study and a brief summary of the results.

1. INTRODUCTION

We began the study by performing a comprehensive survey of the literature on redundancy. The survey resulted in selecting ten of the most promising size-reduction techniques for further investigation. These methods were developed by S. Zionts and J. Wallenius, T. Gal, J. Telgen, D. Rubin, T. Mattheiss, D. Klein and S. Holm, H. Williams, A. Boneh, A. Sethi and G. Thompson, and G. Bradley, G. Brown and G. Graves. The first nine methods were programmed by one of us (Lotfi) in FORTRAN to be tested on the computer. The last method (Bradley, Brown and Graves) was mainly concerned with large-scale problems and the substantial reduction in problem size that can be made by exploiting problem structure. Even quite simple tests such as zero rows or singleton columns can lead to large reductions in large-scale problems presented to a problem solver. We presented the method in the volume but did not program it for further testing.

2. PROGRAMMING THE METHODS

The FORTRAN language was used for our programming purposes because of its widespread use and because most of the participants were familiar with it. As a result, when we faced difficulties in interpretation, it was easier to consult with the participant to overcome the difficulties.

In programming the methods, we tried to minimize the effects of any programming bias. This was done by using common subroutines where possible (i.e., certain methods had one or more steps in common, e.g., the simplex pivot). Moreover, we utilized a relative zero (10^{-8}) in arithmetic operations to minimize the effects of round-off errors. Any number smaller than 10^{-8} was set to zero. In this manner, disagreements among methods on the status of a given constraint (e.g., weakly redundant, strongly redundant and nonredundant) were eliminated. We also provided an initial feasible solution for all of the methods which needed one. The computational effort used to obtain such a solution was not included in evaluating the performance of each method.

3. THE EXPERIMENTAL DESIGN

In order to evaluate the relative efficiencies of the selected methods, two types of LP problems were used. First we solved thirty randomly generated problems by each method and collected relevant statistics. Then we used thirteen structured and real-world LP problems in order to verify our conclusions drawn from randomly generated problems.

These requirement are intended as guidelines to the screening committee but are not to be viewed as binding when work of exceptional merit comes close to satisfying them.

Frequency & Amount of the Award:

The award will be presented, triennially at the Awards Ceremony to the Imternational Symposium on Mathematical Programming Sponsored by the Mathematical Programming Society. The prize for 1985 consists of $500.00 and a certificate.

Judgement Criteria:

Nominations will be judged on the following criteria:

1. Magnitude of the contribution to the advancement of computational and experimental mathematical programming.

2. Originality of ideas and methods.

3. Degree to which unification or simplication of existing methodologies is achieved.

4. Clarify and excellence of exposition.

The Awards Committee:

The screening process will be carried out by a committee of four to eight people appointed by the chairman of COAL at least one year prior to the awarding of the prize, with the review and approval of the chairman of MPS.

Each committee member will read all nonimations and provide to the chairman of the committee their assessment based on the right to determine that there will be no prize given for that meeting.

Nominations:

Nominations must be in writing and include the title(s) of the paper(s) or book, the author(s), the place and date of publication and four copies of the material. Supporting justification and any supplementary materials are welcom but not mandatory. All nominations must be received at least one year prior to the awards date. The screening committee reserves the right to request further supporting materials from the nominees.

MPS Prize in Experimental and Computational Mathematical Programming

Purpose:

To recognize and reward excellence in experimental and computational mathematical programming.

Eligibility:

To be eligible a paper or a book must meet the following requirements:

1. It must be on computational mathematical programming. The topics to be considered include:

   a. experimental evaluations of one or more mathematical programming algorithms,

   b. the development of new methods for empirical testing of mathematical programming techniques (e.g., design of a computational experiment, identification of new performance measures, methods for reducing the cost of empirical testing),

   c. the development of quality mathematical programming software (i.e. well-documented code capable of obtaining solutions to some important class of MP problems) coupled with documentation of the application of the software to this class of problems, or

   d. the development of a new computational method that improves the state-of-the-art in computer implementations of MP algorithms (e.g., a faster update formula, or a scheme for storing networks that speeds traversal) coupled with documentation of the experiment which showed the improvement.

2. It must have appeared in the open literature. If software, it must be available in a language acceptable to the screening committee.

3. Documenation must be written in the English language (translations of material published in other languages is acceptable).

4. The publication date must be within the three year period prior to the year in which the triennial meeting of the MPS is held. (Papers elligible for the 1985 prize must have been published within the years 1982 through 1984. The next prize will consider papers published within the three year period beginning 1985.)

The randomly generated problems were generated by using a modified version of O'Neill and Layman's method [3]. The thirty problems consisted of six sets of problems with different characteristics as follows:

	Problem* Set	Constraints	Variables**	Degeneracy	Redundancy
Level 1	1	10	10	None	None
	2	10	10	50%	None
Level 2	3	20	10	None	50%
	4	20	10	50%	50%
Level 3	5	20	10	None	None
	6	20	10	50%	None

* Each set contained 5 problems.
** Excluding slack variables.

The above design enabled us to make pairwise comparisons among different sets of problems for their special characteristics. For example, we compared the performance of a given method on degenerate problems versus nondegenerate problems either for small problems such as sets 1 versus 2 or larger problems such as sets 5 versus 6.

The structured problems were collected from various sources and ranged in size from 14-constraints, 12-variables to 96-constraints, 70-variables.

4. SUMMARY OF THE RESULTS

The methods of Zionts and Wallenius, Telgen, Gal, and Rubin are based on the same basic principle. That is, all of these methods classify constraints by minimizing the associated slack variable. In doing so, these methods perform various tests on the signs of the contracted-simplex tableau. We refer to these methods appropriately, as the sign test methods.

The main objective of the sign test methods is to identify redundant constraints. However, when applied to the dual problem, they may be used to identify nonbinding constraints and/or extraneous variables. We evaluated the sign test methods for their main objective. In general, these methods performed the same on smaller-sized problems. On larger-sized and structured problems Zionts and Wallenius' method generally performed best although there were some problems in which Gal's and Telgen's method did very well. The latter two methods' better performance was most evident on certain structured problems.

The random direction method of Boneh was rather disappointing. The coordinate direction option (suggested by Telgen and implemented and tested by Boneh) was much more efficient than the random direction option. However, when compared to the sign test methods it did not prove any better. Its strength lies in its ability to handle nonlinear problems while the sign test methods are only applicable to LP problems.

Examples of research which would not quality for this prize:

1. Any proprietary software.
2. Unsubstantiated claims of performance of MP algorithms or software.
3. A collection of test problems without its application to some experimental or application effort.
4. The abstract design of an experiment without any examples of its usefulness to some important class of MP problems.

PROPOSAL FOR A PRIZE IN COMPUTATIONAL MATHEMATICAL PROGRAMMING

The following is a proposal for a prize to be offered in computational mathematical programming. The Committee on Algorithms is proposing to the Mathematical Programming Society that such a prize be given to acknowledge that the field of mathematical programming rests fundamentally on the empirical performance of algorithms. This proposal is currently in the drafting stage and is presented here for your review. We would likt to receive any suggestions you might have by April 1, 1983 so that the Committee can, where appropriate, incorporate these suggestions into the proposal which will be submitted to Council by May 1, 1983. Please send these suggestions to Karla Hoffman, Center for Applied Mathematics, National Bureau of Standards, Washington, D.C. 20234.

To clarify what the committee had in mind when we proposed this prize, I present a brief list of examples of research topics which might be considered such a prize committee as well as research not considered relevant for this prize. These lists follow. Note that these lists are merely examples to clarify the intentions of those drafting the proposal and are not all-inclusive.

Examples of research which might be nominated:

1. Computional investigations which contribute to systematic emperical knowledge or to improved understanding of the convergence characteristics of mathematical programming algorithms.
2. Computational experimentation showing the capabilities of a new MP method over previous techniques used to solve that class of problems.
3. Testing of a variety of well-used MP Software packages in a way which unambiguously indicated the superority of certain techniques.
4. A comparison of "artifical" and "real-world" test problems that clearly reveals their similarity and differences.
5. The development of methods for reducing the cost of emperical testing of classes of MP software.

Mattheiss' method was based on enumerating the entire set of extreme points (of an embedded polytope). The redundant constraints are never binding at any extreme point. This approach required a large storage space and did not perform as well as the sign test methods. Simply put, the embedded polytopes generally contained a large number of extreme points.

The methods of Klein and Holm, and Williams attempted to identify nonbinding constraints and extraneous variables. Klein and Holm's method solved a problem with special structure (problems with nonnegative constraint matrices). For this method we used a special set of test problems. We then compared the results of Klein and Holm's method with those of Zionts and Wallenius' method after eliminating certain obvious implicit equalities and extraneous variables. Though Klein and Holm's method took less time and generally fewer iterations, it did not identify all redundancy or extraneity. The method uses relatively little time so that it could be used for "quick" reductions. Williams' method performed best on problems with a high degree of redundancy and degeneracy. But its overall performance when compared with the sign test methods was not very impressive. The significant factor in reducing its potential were the lack of tight variable bounds.

Thompson and Sethi's "noncandidate constraint" method was compared with the regular simplex method. The noncandidate constraint method outperformed the regular simplex method on problems with high degrees of redundancy and degeneracy. Though the authors had suggested that the method's efficiency improves with increase in size, we could not verify this. Thompson and Sethi have made improvements to their approach and also tested them on special problem structures since our study was completed. Their work looks quite promising.

5. OTHER DEVELOPMENTS

In studying the various size-reduction techniques, we discovered many possibilities for improvements and extensions. We selected some of the more promising ideas and tested them. Our first improvement, called the extended sign test method, was the result of combining and modifying the sign test methods. As we had anticipated, the extended sign test method outperformed the sign test methods consistently.

We then combined the extended sign test method with the coordinate direction method. This hybrid method outperformed the coordinate direction method. However, it performed the same as the extended sign test method.

Our last extension was to apply the extended sign test method to both the primal and the dual problems while solving the problem. This approach enabled us to identify redundant constraints as well as nonbinding constraints and extraneous variables. We eliminated such variables and constraints as soon as they were identified thereby reducing the size of the problem. This approach proved rather impressive in special cases. In particular, on larger problems

Besides these administrative activities, there were many well attended technical sessions covering topics related to computational and experimental mathematical programming. In all respects, the 11th MTS Symposium on Mathematical Programming helped advance the objectives of COAL.

COAL-RELATED ACTIVITIES AT THE 11TH MPS SYMPOSIUM IN BONN

At the council meeting of the Mathematical Programming Society the following COAL-related actions were taken:

1. Approval by concensus of the Council of the acceptance of the resignations of three members of the Committee and the appointment of five new members. A list of the current members of COAL can be found elsewhere in the newsletter.
2. Acceptance of Ric Jackson's resignation as chairman and of Karla Hoffman's appointment as Chairman.
3. Acceptance of Jan Telgen's appointment as Editor of the COAL Newsletter.
4. The Society agreed to undertake additional financial support of the COAL newsletter. The Council has authorized a budget of no more than 1500 dollars per year for this support.
5. After much discussion about the possibility of establishing a COAL Prize for excellence in experimental and computation mathematical programming, it was decided that a draft proposal would be sent to council members for comment. COAL would then submit to council a final proposal at a later date. A copy of the current version of the proposal can be found elsewhere in the newsletter. We welcome all comments and discussion about this proposal.

On Tuesday evening, COAL held their open business meeting in which the decisions of MPS Council were announced. In addition, it was agreed at that meeting that:

1. COAL would organize our third international conference on experimental optimization to be held in Europe in the summer of 1984. Klaus Schitthowski agreed to take responsibility fo organizing this meeting. It was agreed that the meeting would be held either directly before or directly after the TIMS meeting which will take place in Denmark that summer.
2. COAL would begin preparing a catalog of test problems in mathematical programming. Procedures for publishing and distributing this library would be determined.
3. A majority of those at the COAL business meeting approved of the idea of establishing a prize in experimental mathematical programming. Therefore, the Committee would continue its efforts in this area.

containing a high degree of redundancy and degeneracy, our extension outperformed the regular simplex method by a wide margin.

In the course of study, several other ideas of improvements and extensions were developed. Many of these ideas warrant further study. The reader may refer to Lotfi [2] or to Karwan, Lotfi, Telgen, and Zionts [1] for full details.

REFERENCES

[1] Karwan, M. H., Lotfi, V., Telgen, J., and Zionts, S., Redundancy in Mathematical Programming, volume under print by Springer-Verlag, Berlin, 1983.

[2] Lotfi, V., "A Study of Size-Reduction Techniques in Linear Programming," Doctoral Dissertation, State University of New York at Buffalo, 1981.

[3] O'Neill, R. P. and Layman, C. H., "A Study of the Effect of LP Parameters on Algorithm Performance," Computers and Mathematical Programming, National Bureau of Standards, 1978.

Calendar of mathematical programming meetings as of 15 December 1982

Maintained by the Mathematical Programming Society (MPS)

This Calendar lists meetings specializing in mathematical programming or one of its subfields in the general area of optimization and applications, whether or not the Society is involved in the meeting. (These meetings are not necessarily "open".) Any one knowing of a forthcoming meeting not listed here is urged to inform Dr. Philip Wolfe, IBM Research 33-221, POB 218, Yorktown Heights, NY 10598, U.S.A; Telephone 914-945-1642, Telex 137456.

Some of these meetings are sponsored by the Society as part of its world-wide support of activity in mathematical programming. Under certain guidelines the Society can offer publicity, mailing lists and labels, and the loan of money to the organizers of a qualified meeting. For further information address the Treasurer of the Society, Dr. A. C. Williams, Mobil Corporation, 150 East 42d Street, New York, New York 10017, U.S.A.; Telephone 212-883-7678.

Substantial portions of meetings of other societies such as SIAM, TIMS, and the many national OR societies are devoted to mathematical programming, and their schedules should be consulted.

1983

March 28-31: NETFLOW83, International Workshop on Network Flow Optimization Theory and Practice, Pisa, Italy. Contact: Dr. Claudio Sandi, IBM Scientific Center, Via S. Maria 67, 56100 Pisa, Italy; telephone 50-47383.

April 5-10: 15th Conference "Mathematical Optimization", Sellin/Rügen, German Democratic Republic. Contact: Dr. R. Hansel, Sektion Mathematik, Humboldt-Universität zu Berlin, 1086 Berlin, German Democratic Republic; telephone 203 2239.

May 13-14: "Optimization Days", Campus de la Université de Montréal, Canada. Contact: Professor Michael Polis, École Polytechnique de Montréal, C.P. 6079, Succ. "A", Montréal, Québec, Canada H3C 3A7; telephone 514-344-4884, Telex 05-24146 BIBPOLYTEC. Sponsored by ACFAS, CAMS, IEEE, and the MPS.

May 16-18: "Fifth Symposium on Mathematical Programming with Data Perturbations", The George Washington University, Washington, D.C., U.S.A. Contact: Professor Anthony V. Fiacco, Department of Operations Research, School of Engineering and Applied Science, The George Washington University, Washington, D.C. 20052, U.S.A.; telephone 202-676-7511. Deadline for abstracts, 1 March 1983.

June 20-21: IFAC Workshop on Applications of Nonlinear Programming to Optimization and Control, San Francisco, California, U.S.A. Abstract deadline February 1, 1983. Contact: Herbert E. Rauch, Lockheed 52-56/205, Palo Alto Research Laboratory, 3251 Hanover Street, Palo Alto, CA 94304, U.S.A.; Telephone 415-493-4411, Extension 45677.

July 4-15: Summer School on Combinatorial Optimisation, Dublin. Contact: Secretariat, Summer School on Combinatorial Optimisation, National Institute for Higher Education, Glasnevin, Dublin 9, Ireland.

July 11-15: 3d IFAC/IFORS Symposium "Large Scale Systems: Theory and Applications", Warsaw. Deadline for abstracts, 15 February 1982. Contact: Dr. Z. Nahorski, 3d IFAC/IFORS LSSTA, Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warszawa, Poland; Telex 812397 ibs pl, Telephones 364103, 368150.

July 25-29: 11th IFIP Conference on System Modelling and Optimization, Copenhagen, Denmark. Deadline for abstracts, 31 December 1982. Contact: Professor P. Thoft-Christensen, Institute of Building Technology and Structural Engineering, Aalborg University Center, P.O. Box 159, DK-9100 Aalborg, Denmark.

We have also begun plans for a NATO-sponsored Summer School on Experimental Mathematical Programming to be held in the summer of 1984. Klaus Schittkowski is coordinator of this symposium. At this meeting, we hope to bring together both developers of MP algorithms, developers of MP software, and users of MP codes in order to expand the communication among these groups and to better assess likely avenues for future applications of MP software.

Finally, we hope to prepare and publish a catalog of test problems in mathematical programming.

These above-mentioned tasks represent the major efforts of COAL in the coming months. With your help, we will be successful in each of these.

MESSAGE FROM THE CHAIRMAN

I am delighted and suprised that in the past few years the number of persons receiving this newsletter has increased from an original mailing list of 60 to one now totalling over 900 names. This newsletter was always planned as an informal outlet for communication about research in computational mathematical programming and the response indicates to me that there is a broad community who is interested in the results of that research. Beginning with this issue, COAL has a new newsletter editor. I would like to take this opportunity to ask that you help Jan by sending him news of your research, of meetings, of notices of talks, information about software and/or test problems and suggestions about what you would like to see in future issues of the newsletter.

Since this is my first message as chairman, I think it is an appropriate time for me to explain that the Committee on Algorithms is composed of only 14 people. Without the help of many "friends" of COAL, the achievements of the committee would not have been possible. I thank you for that assistance and hope that it will continue and even grow in the next few years.

We are now in the process of reviewing our past objectives and considering what future directions the committee should take. Originally, COAL was concerned solely with the testing of Mathematical Programming Software. However, recently many of the memebers and "friends" have suggested that our objectives should be broadened to include other computational aspects of Mathematical Programming. I would welcome your views of this issue.

Our activities in the near future include sessions at upcoming ORSA/TIMS meetings and at other international society meetings. Included in this newsletter is a proposal for a prize to be awarded for research in computational mathematical programming. If the Council of the Mathematical Programming Society endorces this idea, the first award will be presented at the 12th Mathematical Programmaing Symposium in 1985.

EDITOR'S COLUMN

It is quite common for newly appointed editors taking over a flourishing journal to state that they are facing an extremely difficult, if not impossible task. The evidence to support this statement consists of noting that:

- the leaving editor did such a great job and
- the journal is doing well.

These facts can be noted for the COAL Newsletter as well. While they provide me with an ideal assist to create an a priori excuse for my performance as editor of the COAL Newsletter, I want to use them to point out something else: Karla Hoffman's contribution to COAL and this Newsletter. She had to bring the Newsletter into existence, and she did more than that. Under Karla's editorship the COAL Newsletter grew out to be a major factor in the very existence of the Committee on Algorithms. She did a marvellous job!

I'll try to put in as much creativeness, perseverence and labour as Karla did as an editor. And I am very glad I won't be alone. Starting with the next issue of the COAL Newsletter, Bob Meyer will be acting as co-editor for this Newsletter. Bob will be the contact for North American contributions to the Newsletter and he will have a special eye on contributions relating to Networks and NLP. I look forward to a fruitful cooperation.

Regarding the editorial policy for the Newsletter I think it is doing well considering the function it was meant for at its conception. Now, after 5 years of operation on this basis, it is appropriate to reconsider its function in view of new developments. These developments include the origination of OPTIMA (the other Newsletter from the Mathematical Programming Society) and the length of the Mailing List in combination with the increasing costs of mailing.
Since MPS is the only source of money for the COAL Newsletter and non-MPS members are receiving the Newsletter too (free of charge), it is only natural to discuss the role of the COAL Newsletter in the context of the other publications of MPS. Until this discussion is finalized the editorial policy for the COAL Newsletter will not be changed.

COMMITTEE ON ALGORITHMS of the
MATHEMATICAL PROGRAMMING SOCIETY

CHAIRMAN

Karla L. Hoffman
Center for Applied Mathematics
National Bureau of Standards
Washington, D.C. 20234
USA

Harlan P. Crowder
IBM Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Richard H. F. Jackson
Center for Applied Mathematics
National Bureau of Standards
Washington, D.C. 20234
USA

Leon S. Lasdon
Department of General Business
School of Business Administration
Austin, TX 78712

Claude LeMarechal
INRIA Domaine deVolvceau
Requen Ct. BP105
78150 Le Chesnay
FRANCE

Robert R. Meyer
University of Wisconsin
Computer Sciences Dept.
1210 W. Dayton St.
Madison, WI 53706

Richard P. O'Neill
EI 622, rm. 4447
Division of Oil and Gas Analysis
Dept. of Energy
Washington, D.C. 20461

EDITOR OF THE NEWSLETTER

Jan Telgen
RABOBANK NEDERLAND
Laan van Eikenstein 9 (ZL-G-170)
3705 AR ZEIST
The Nederlands

Susan S. Powell
The London School of Economics
and Political Science
Houghton St.
London WC2A 2AE
United Kingdom

Ken Ragsdell
School of Mechanical Engineering
Purdue University
West Lafayette, IN 47907

Ronald Rardin
School of Ind. & Systems Engineering
Georgia Inst. of Technology
Atlanta, GA 30332

Patsy Saunders
Center for Applied Mathematics
National Bureau of Standards
Washington, D.C. 20234

Klaus Schittkowski
Institut fur Angewandte Mathematik und Statistik
Universitat Wurzburg
D-87 Wurzburg
West Germany

Robert Schnabel
University of Colorado
Dept. of Computer Science
Boulder, CO 80309

EX OFFICIO MEMBERS

J. Abadie, Chairman, MPS
29, Boulevard Edgar-Quintet
75014 Paris, France

Philip Wolfe, VIce Chairman, MPS
IBM Research 33-2
P.O. Box 218
Yorktown Heights, NY 16598

A.C. Williams, MPS Exec. Comm.
Mobil Oil Co. Technical Center
P.O. Box 1025
Princeton, NJ 08540

COAL OBJECTIVES

The Committee on Algorithms is involved in computational developments in mathematical programming. There are three major goals: (1) ensuring a suitable basis for comparing algorithms, (2) acting as a focal point for computer programs that are available for general calculations and for test problems, and (3) encouraging those who distribute programs to meet certain standards of portability, testing, ease of use and documentation.

NEWSLETTER OBJECTIVE

The newsletter's primary objective i
for the Friends of COAL. Through an ir
opinions, members have an opportunity
experiences. To date, our profession h
clear understanding on the issues of h
should be carried out, how the results
be presented in the literature, or ho
algorithms should be properly evaluat
issues will be addressed in the newsl [redacted]

Mathematical Programming Society
Committee on Algorithms Newsletter

8 MAART 1983 Jan Telgen, Editor

Contents:

Dr. JAN TELGEN
RABOBANK NEDERLAND
LAAN VAN EIKENSTEIN 9
3705 AR ZEIST
THE NETHERLANDS

PRINCE GEORGES, MD
PM
JUN 15
1983

Robert Millikan
37c
USA

FIRST CLASS MAIL

[redacted]